

## Quality of Service (QoS) Specification and Management for XML Web Services

**Dr. Vladimir Tosic**

Dep. of Computer Science  
Univ. of Western Ontario

London, Ontario, Canada

E-mail: [vladat@computer.org](mailto:vladat@computer.org)

<http://elab.njit.edu/vladimir/>

**Prof. Patrick C.K. Hung**

Faculty of Business and IT  
Univ. of Ontario Institute of  
Technology - UOIT

Oshawa, Ontario, Canada

Patrick.Hung@uoit.ca

<http://www.cs.ust.hk/~cshck/>

Slide  
2 of 80

### Tutorial Goals

- ⌘ Explain that QoS specification and management are **crucial** for achieving the vision of service-oriented computing (publish-find-bind model)
- ⌘ Inform that there has been **a lot of** academic and industrial work in the area
- ⌘ **Summarize and analyze** the main past results
- ⌘ **List and discuss** open issues, such as **standardization** of specification and management of QoS for Web services (WSes)
- ⌘ **Provide a foundation** for future research and/or decision-making by the participants

Slide  
3 of 80

### Presentation Outline

- I. Introduction: **Importance of QoS** for WS
- II. Approaches to **specification of QoS** for WS
- III. **Languages** for specification of QoS for WS
  - ⌘ 10 minute break
- IV. Approaches to **management of QoS** for WS
- V. **Research and industrial tools** for management of QoS for WS
- VI. Summary: **Past results and open issues**
  - ⌘ Answers to questions and **discussion**

Module  
1 of VI

Slide  
4 of 80

### Module I:

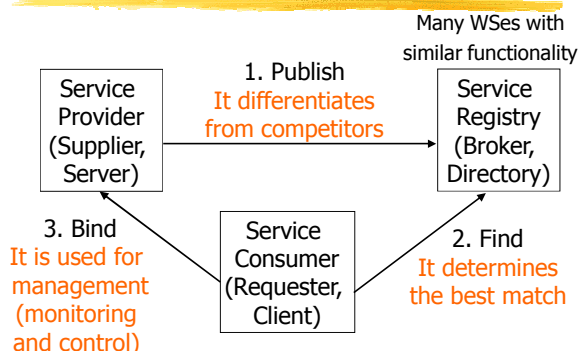
#### Definition of Terms and Importance of Web Service Quality of Service (QoS)

- ⌘ **Functionality/service** = "WHAT operations does the system execute?"
  - ☑ Example: Returns current price for a stock symbol
- ⌘ **Quality of service (QoS)** = "HOW WELL the system performs its operations?"
  - ☑ Examples: Average response time is 2 seconds, availability in the last 24 hours is 99%, ...
  - ☑ Synonyms: non-functional, extra-functional, 'ilities'
  - ☑ Price and security information sometimes included
  - ☑ QoS exists even when not specified or measured

Module  
1 of VI

Slide  
5 of 80

#### The Need for QoS Information in the Service-Oriented Architecture (SOA)



Module  
1 of VI

Slide  
6 of 80

#### Definition of Management - Monitoring

- ⌘ **Management = monitoring and control**
  - ☑ **Run-time** (and some deployment-time) activities
- ⌘ **Monitoring** determines state of the system:
  - ☑ Measurement or calculation of QoS metrics (measures of QoS): response time, availability, ...
  - ☑ Evaluation of conditions (requirements or guarantees): response time < 2 seconds, ...
  - ☑ Accounting of invoked operations, consumed resources, measured/calculated QoS metrics, evaluated conditions, taken control actions, billed prices/penalties, ...

Module 1 of VI Slide 7 of 80

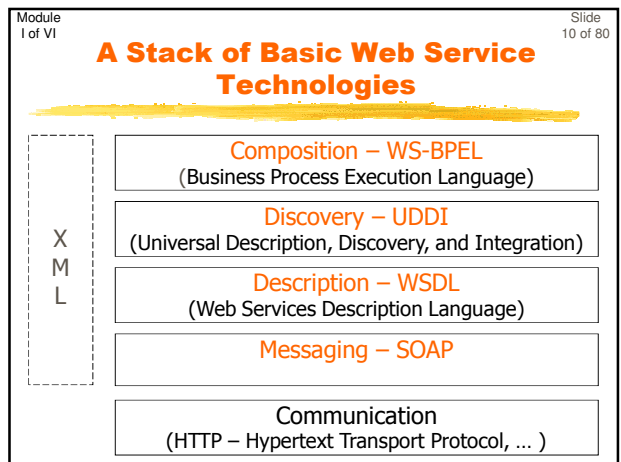
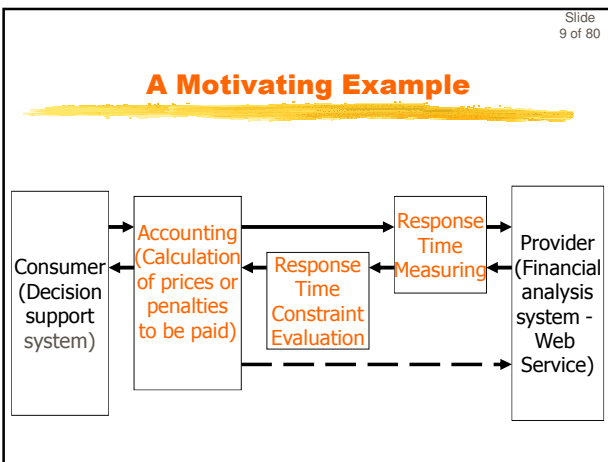
## Definition of Management - Control

- ⌘ **Control** tries to ensure that the managed system is always in its desired state:
  - ☒ Starting/stopping the system or its components
  - ☒ (Re-)Configuration of the system: setting thread priorities, re-composition of Web services, ...
  - ☒ (Re-)Allocation of resources: assigning processing time to requests from different consumers, ...
  - ☒ Billing of prices or penalties: penalty for not meeting guaranteed response time is US\$1.00, ...
  - ☒ Modification of requirements or guarantees
  - ☒ Notification of human administrators

Module 1 of VI Slide 8 of 80

## Benefits of QoS Management

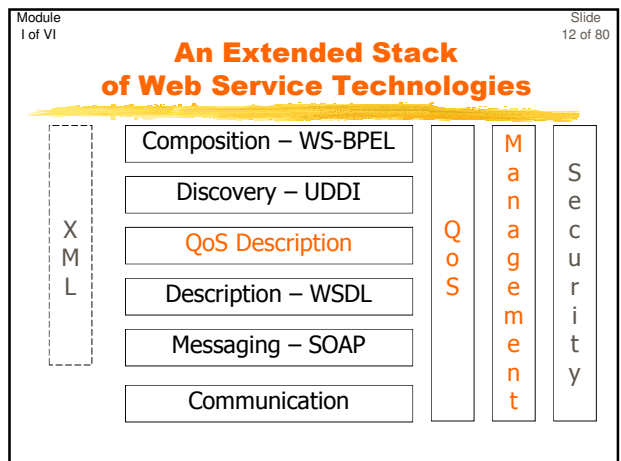
- ⌘ 5 functional areas of system/network management (FCAPS): Fault, Configuration, Accounting, **Performance**, and Security
- ⌘ **QoS (performance) management** helps to:
  - ☒ ensure correct operation,
  - ☒ attain or surpass guaranteed QoS,
  - ☒ discover and fix problems,
  - ☒ accommodate change,
  - ☒ balance price/performance ratios,
  - ☒ maximize profits, ...



Module 1 of VI Slide 11 of 80

## The Place of QoS in the Stack of Web Service Technologies

- ⌘ Many additional technologies appeared
- ⌘ There are disagreements about the contents of the stack of Web service technologies
- ⌘ **Basic Web service technologies (SOAP, WSDL, UDDI, WS-BPEL) do not address QoS specification and management**
- ⌘ **Two approaches** to adding QoS specification and management to the stack
  1. Crosscutting different layers
  2. Adding a new layer



Module I of VI Slide 13 of 80

## Examples of QoS Issues at Different Layers in the Stack

- ⌘ SOAP: "Exactly once" delivery, compression of SOAP messages, ... (not the focus of this tutorial)
- ⌘ WSDL: Describing QoS in addition to functionality (can be a separate layer in the stack), management of QoS of one WS, ...
- ⌘ UDDI: Discovery/selection of Web services with particular functionality and QoS, ...
- ⌘ WS-BPEL: Describing QoS dependencies between WSEs in a composition, management of QoS of the whole composition, ...

Module I of VI Slide 14 of 80

## What Has to Be Developed for WS QoS Specification and Management?

1. Well-defined (ideally: standardized) **formats** for specification of WS QoS information
2. Many WS QoS-related **algorithms & protocols**
  - ☒ Very diverse: selection of WSEs using QoS info, negotiation of QoS of a WS, monitoring of QoS, exchange of run-time QoS info, control to achieve QoS guarantees, adaptation to changes in QoS, ...
  - ☒ We will discuss them in the context of specification formats and/or management tools
3. WS QoS **management infrastructures/tools**

Module II of VI Slide 15 of 80

## Module II: Overview of Approaches to WS QoS Specification

- ⌘ QoS specification = description of what/where/when/how to monitor & control
- ⌘ QoS info = descriptions & monitored values
- ⌘ Classification of QoS specification **approaches**:
  1. **Implicit** – built into the implementation (not flexible)
  2. **Contracts** – formal agreements (for QoS, billing, ...)
    - ☒ Service Level Agreements (SLAs)
    - ☒ Classes of service – a special type of SLAs
  3. **Policies** – high-level operation & management goals and/or rules (for security, QoS, billing, ...)

Module II of VI Slide 16 of 80

## Contract

- ⌘ **Contract** = (binding and enforceable) formal agreement between two or more parties
- ⌘ Defines **requirements & guarantees** of parties
  - ☒ Can be used in monitoring and control
- ⌘ Contracts enable not only **QoS description**, but also **QoS differentiation**
  - ☒ Different consumers can have different contracts
- ⌘ Apart from QoS info, a contract can contain **other information** (e.g., prices/penalties)
  - ☒ A WSDL file is a contract

Module II of VI Slide 17 of 80

## Specification of QoS in Extended WSDL, UDDI, or WS-BPEL Files

- ⌘ **Strengths**:
  - ☒ The extensions can be relatively simple
  - ☒ QoS discovery related to Web service discovery
- ⌘ **Weaknesses**:
  - ☒ QoS specification language tied to WSDL (UDDI, WS-BPEL) in terms of tools, evolution, ...
  - ☒ Extension mechanisms are limited
  - ☒ **Run-time change of QoS information** requires updates of all affected copies of WSDL (UDDI, WS-BPEL) files, which is complicated

Module II of VI Slide 18 of 80

## Service Level Agreement (SLA)

- ⌘ A special **type of contract** for QoS (and often price/penalty) requirements & guarantees
- ⌘ Many different **formats**, one of them is:
  - ☒ **Parties** (including supporting management parties)
  - ☒ **Service description**
    - ☒ Service operations – describe available operations
    - ☒ SLA parameters – define monitoring of QoS metrics
  - ☒ **Obligations**
    - ☒ **Service Level Objectives (SLOs)** - QoS guarantees
    - ☒ **Action guarantees** - specify what happens if SLOs are met or not met

Module II of VI Slide 19 of 80

### A Simple Example of an SLA

**Parties:** consumer C and provider P

**Service operations:** P has one operation (OP1)  
float getStockPrice(String stockName)

**SLA parameters:** (RT-OP1-C) Response time of operation OP1 measured at consumer C by consumer C

**SLOs:** (SLO1) For every OP1 invocation by C, RT-OP1-C will be less than or equal to 2 seconds

**Action guarantees:** (AG1) If SLO1 was met, C pays P price of US\$0.20 per invocation;  
(AG2) If SLO1 was not met, P pays C penalty of US\$0.10 per invocation

Module II of VI Slide 20 of 80

### Service Level Agreement (SLA) – Strength and Weaknesses

- ⌘ **Strengths:**
  - ☒ Formal contract specification of QoS and related management aspects
  - ☒ Widely used in computing and communications systems (and now also for WSeS)
- ⌘ **Weaknesses:**
  - ☒ Negotiation of custom-made SLAs can require complex analysis of offers and generation of counter-offers (can be alleviated by using templates)
  - ☒ Management of many concurrent custom-made SLAs can be complex & with high run-time overhead
  - ☒ Cannot be used for QoS-enabled WS selection

Module II of VI Slide 21 of 80

### Class of Service

- ⌘ A special type of SLA that is not custom-made, but predefined & reusable (anonymous)
  - ☒ 1 provider can offer many classes of service that refer to the same functionality, but differ in QoS
  - ☒ 1 class of service can be used by many consumers
  - ☒ Simple selection instead of complex negotiation
  - ☒ Classes of service already checked for consistency
  - ☒ Strengths: Usable for QoS-enabled WS selection, no complex negotiation, simpler management, lower run-time overhead, faster adaptation
  - ☒ Weakness: Discrete differentiation - limited choice

Module II of VI Slide 22 of 80

### Policies – High-Level Goals and/or Rules

- ⌘ One classification of policy types:
  - ☒ Action: Describe what should happen - "If-Then" rules ("If response time of operation A is greater than 2 sec, provider pays penalty of US\$0.10")
  - ☒ Goal: Describe desired state ("Response time of operation A is less than or equal to 2 sec")
  - ☒ Utility: Quantify "goodness" of a particular state ("Add to the goodness measure [2 sec - response time of operation A] \* 10 units") - rarely used
- ⌘ SLAs vs. policies: SLOs can be viewed as goal policies, action guarantees as action policies

Module II of VI Slide 23 of 80

### QoS Specification Topics Present in All Approaches

- ⌘ Where are QoS metrics defined?
  - ☒ There are no standard QoS metrics - use, names, and definitions vary! Example: 'response time' can have at least 2 different meanings! 4 approaches:
    1. Nowhere (implicit meaning) - not precise
    2. In the QoS language grammar - not flexible
    3. In QoS specification files (e.g., SLAs) - not reusable
    4. In external reusable ontologies (definition files)
      - ☒ Other ontologies can define measurement units
- ⌘ For practical use, QoS specification languages must be accompanied by appropriate tools!

Module III of VI Slide 24 of 80

### Why Not Reusing QoS Specification Languages from Other Areas?

- ⌘ Many existing QoS specification languages in multimedia (HQML, ...), distributed objects (QML, QDL, QIDL, ...), and other areas
- ⌘ Can not be directly re-used because of:
  - ☒ Incompatibility with WS standards (e.g., WSDL)
  - ☒ Heterogeneity of WS implementations and interactions styles used (asynchronous & synchronous; document-based & RPC; ...)
  - ☒ Characteristics of WS compositions: business-to-business, Internet scale, dynamism, automatism, ...

Module III of VI Slide 25 of 80

## Module III: Overview of XML Languages for WS QoS Specification

- ⌘ Classified based on the main concept:
  1. **SLA**: WSLA, [SahaiEtAl2002] (WSML), SLAng
  2. **Class of service**: WSOL, WS-QoS, DAML-QoS
  3. **General contract**: OWL-S, WS-Agreement
  4. **Extension of UDDI/BPEL**: UDDIe, [McGregor2003]
  5. **Policy**: WS-Policy
  6. **Manageability capability**: WSDM
- ⌘ There are some other languages and formats, but these are probably best representatives

Module III of VI Slide 26 of 80

## Web Service Level Agreement (WSLA) – Overview

from IBM Research: H. Ludwig, A. Keller, A. Dan, ...

- ⌘ **QoS language & management infrastructure**
- ⌘ Compatible with, but not restricted to WSEs
- ⌘ **Custom-made SLAs** that have 3 parts:
  1. **Parties** (signatory & supporting) and their management operations
  2. **Service definitions** – service objects (e.g., WSDL operations) and their monitored properties (**QoS metrics**, SLA parameters, schedules, triggers, ...)
  3. **Obligations** – SLOs and action guarantees

Module III of VI Slide 27 of 80

## Web Service Level Agreement (WSLA) - Language Details

- ☒ **SLA parameter** - monitored property; contains 1 QoS metric & extra info for exchange of values
- ☒ **QoS metric** – defines where & how to measure or calculate; can be reused across SLA parameters
- ☒ An **SLO** contains: **evaluated Boolean expression** (limits values of SLA parameters), obliged party, validity periods, evaluation event or schedule
- ☒ An **action guarantee** contains: precondition expression, evaluation event or schedule, **action to be taken**, obliged party, execution modality
- ☒ Reusability: SLA templates, metric macros, ...

Module III of VI Slide 28 of 80

## Web Service Level Agreement (WSLA) – Strengths and Weaknesses

- ⌘ **Strengths** (significantly outweigh weaknesses):
  - ☒ **Detailed and precise** description of QoS monitoring and control
  - ☒ Several **tools** for SLA creation, deployment, and compliance monitoring – were distributed with IBM's Emerging Technologies Toolkit (ETTK)
  - ☒ The **most used** WS QoS specification language
- ⌘ **Weaknesses** (mainly due to custom-made SLAs):
  - ☒ QoS metrics defined within SLAs
  - ☒ Can not be used for QoS-enabled WS selection
  - ☒ High overhead of supporting custom-made SLAs

Module III of VI Slide 29 of 80

## HP's Web Service SLA Language ([SahaiEtAl2002])

from HP Labs: A. Sahai, A. Durante, V. Machiraju, ...

- ☒ a.k.a. Web Service Management Language (WSML)
- ☒ for **Web Services Management Network (WSMN)**
- ⌘ **Custom-made SLAs** - Each contains: validity period, parties, and multiple SLOs
- ⌘ An **SLO** has: validity times & multiple clauses
- ⌘ A **clause** defines: monitored items (and places of their monitoring), evaluation times, samples, evaluation function, action to take
- ⌘ Similar to WSLA, but not as widely used

Module III of VI Slide 30 of 80

## SLAng

from UC London: D.D. Lamanna, J. Skene, W. Emmerich

- ⌘ For **custom-made SLAs** for ASPs (WSEs) and underlying resources (horizontal/vertical SLAs)
- ⌘ **Weaknesses**: 1) Still "work in progress"; 2) **No management infrastructure**; 3) QoS metrics defined in language grammar – not flexible, hard to extend
- ⌘ **Strengths**: 1) Formally and very precisely defined semantics of the language and built-in QoS metrics; 2) Use meta-modeling to define language's syntax and semantics; 3) SLA conformance checker generated from the language definition

Module III of VI Slide 31 of 80

## Web Service Offerings Language (WSOL) – Overview

from Carleton Univ.: V. Tasic, K. Patel, B. Pagurek, ...

- ☒ plus **Web Service Offerings Infrastructure (WSOI)**
- ☒ **Weakness:** parser available, but not a complete compiler generating monitoring code
- ☒ **Classes of service** and their **relationships**
- ☒ **Multiple categories of constraint:** functional, QoS (including periodic), access rights, context
  - ☒ Also: various prices and monetary penalties
- ☒ QoS metrics defined in external ontologies
- ☒ Its goal was relatively low run-time overhead

Module III of VI Slide 32 of 80

## Web Service Offerings Language (WSOL) – Language Constructs

- ☒ **Service Offering (SO)** - for a class of service
  - ☒ **constraints** - expressions (Boolean, arithmetic, ...)
  - ☒ **statements** (prices, penalties, management responsibility, disconnection handling)
  - ☒ **reusability constructs** (extension, constraint groups, inclusion, instantiation of constraint group templates, ...)
- ☒ **Service Offerings Dynamic Relationships (SODRs)** – if a set of constraints from current SO was not met, what is an appropriate replacement SO

Module III of VI Slide 33 of 80

## WSDL and WSOL: An Example

buyStock Web Service

buyStock.wsdl

- buyStockPortType
- buyStockOperation
- buyStockRequest
- symbol
- quantity
- buyStockResponse
- totalStockBuyingCost
- ...
- buyStockBinding
- buyStockPort

buyStock.wsol

ServiceOffering1

accounting party: provider

subscription (1 day): 0.5\$

Constraint1: precondition domain: buyStockOperation quantity > 0

Constraint2: QoSConstraint domain: buyStockOperation ResponseTime < 0.5 s

ServiceOffering2 ...

Module III of VI Slide 34 of 80

## Web Service Offerings Language (WSOL) – Syntax Example

```
<wsol:serviceOffering name="SO1" service="buyStock:buyStockService" accountingParty="WSOL-SUPPLIERWS">
...
<wsol:instantiate CGTName="CGT2" resService="..." resPortOrPortType="..." resOperation="..." resCGName="CG5">
  <wsol:parmValue name="maxResTime">
    <wsol:numberWithUnitConstant>
      <wsol:value>30</wsol:value>
      <wsol:unit type="QoSMeasOntology:millisecond"/>
    </wsol:numberWithUnitConstant>
  </wsol:parmValue>
</wsol:instantiate>
</wsol:serviceOffering>
```

Module III of VI Slide 35 of 80

## Web Services Quality of Services (WS-QoS)

from Freie Univ. Berlin: M. Tian, A. Gramm, H. Ritter, ...

- ☒ plus **WS selection and monitoring infrastructure**
- ☒ For **classes of service** on the WS level and network level (cross-layer, cross-domain approach)
- ☒ Strength: QoS specified for **both** providers & consumers, so that they can be matched
- ☒ Weakness: **Limited format** (no expressions, ...)
- ☒ Informal definitions of several QoS metrics built into the language grammar, but new ones can be added through WS-QoS ontology

Module III of VI Slide 36 of 80

## Web Ontology Language (OWL) – Services (OWL-S)

from the Semantic Web community

- ☒ Early version was called DAML-S
- ☒ Early version of OWL was called DAML, later DAML+OIL
- ☒ Intended for **WS selection**, not QoS monitoring
- ☒ **"Service profile"** contains comprehensive description of WSeS, including some QoS, functional constraints, prices, ...
- ☒ Can be viewed as a general contract
- ☒ Weaknesses: QoS specification is **very weak** – not nearly enough for monitoring activities

Module III of VI Slide 37 of 80

## DAML-QoS - Overview

from Nanyang Tech.Univ.: C. Zhou, L.T. Chia, B.S. Lee

- ⌘ QoS **complement to DAML-S**; in DAML+OIL
- ⌘ **3 layers** with reusable info for QoS matching:
  1. **QoS profile** (provider/inquiry/template): contains scope, evaluation time; DAML+OIL cardinality is used to represent QoS property constraints; many QoS profiles per 1 DAML-S service profile
  2. **QoS property definition**: contains domain (input, output, ...), range (QoS metric)
  3. **QoS metric** (atomic/complex): monitoring info; some metrics built-in, others can be added

Module III of VI Slide 38 of 80

## DAML-QoS - Discussion

- ⌘ **Strengths:**
  - ☒ Significant improvement over OWL-S (DAML-S)
  - ☒ Reuse of existing Semantic Web tools for reasoning
  - ☒ QoS specified for both providers and consumers
- ⌘ **Weaknesses:**
  - ☒ **No management infrastructure**
  - ☒ **Very hard to understand** – unusual design choices; it is not clear what monitoring information can or cannot be specified and how is this monitored
  - ☒ No support for control activities
  - ☒ Overhead of Semantic Web technologies

Module III of VI Slide 39 of 80

## Web Services Policy Framework (WS-Policy)

from BEA/IBM/Microsoft/SAP – **industry support!**

- ⌘ **General, flexible and extensible, framework** for specification of (security) policies for WSEs
- ⌘ Many good features (e.g., policies can be in or out of WSDL files, some reusability constructs)
- ⌘ **QoS extension is possible, but missing! To add:**
  - ☒ Precise and detailed QoS specification
  - ☒ Contracts/SLAs/classes of service and their static and dynamic relationships
  - ☒ Standardized expression mechanism

Module III of VI Slide 40 of 80

## WS-Agreement

from Global Grid Forum (GGF); industry support (IBM,...)

- ⌘ **General framework** for XML specification of agreements and agreement templates
  - ☒ plus a simple agreement negotiation protocol and run-time agreement monitoring interface
- ⌘ Intended for **multiple domains**, not only WSEs
- ⌘ WS-Agreement allows use of **any language** for the actual contained specifications (including QoS), expressions, QoS metrics, ...
- ⌘ This **flexibility** can produce **incompatibility**

Module III of VI Slide 41 of 80

## WS-Agreement – Agreement Template Structure

- ⌘ **Name**
- ⌘ **Context:** Involved parties (initiator & provider); Expiration time; Template name, Related agreements
- ⌘ **Terms:** Term compositors ExactlyOne/OneOrMore/All
  - ☒ **Service description terms:** Service descriptions, Service references, Service properties
  - ☒ **Guarantee terms:** Service scope, Qualifying condition, **Service level objective (SLO)**, Business value list (Importance, Penalty, Reward, Preference, ...)
- ⌘ **Creation constraints:** item requirements and/or constraints (in some language)

Module III of VI Slide 42 of 80

## Some Simple and Limited Languages - Extensions of UDDI, WSDL, or BPEL

- A. **UDDIe (UDDI Extension)** – Adds to UDDI licensing information and a bag of arbitrary properties that can represent QoS metrics
  - ☒ QoS info also specified in extended WSDL files
- B. Simple **BPEL4WS extension** ([McGregor2003]) with several predefined QoS metrics to monitor and analyze business performance
  - ☒ Monitored QoS metrics and their target values

from Univ. of Western Sydney: C. McGregor

Module III of VI Slide 43 of 80

## Web Services Distributed Management (WSDM)

from OASIS Web Services Distributed Management TC

- ☒ One of the used inputs: WS-Manageability
- ⌘ Two sets of publications:
  - ☒ Management Using Web Services (MUWS) - manageability interfaces of resources exposed as Wses
    - ☒ Manageability capability – set of operations, properties, events, metadata, and other info. Example: "metrics"
  - ☒ Management Of Web Services (MOWS) – WSes managed as resources and described with MUWS
    - ☒ Defines basic metrics, e.g., LastResponseTime
- ⌘ Strengths: Standard management interfaces
- ⌘ Weaknesses: Very limited QoS specification and management support; no specification of guarantees

Module III of VI Slide 44 of 80

## Languages for WS QoS Specification – Concluding Remarks

- ⌘ There are many languages; very different
- ⌘ All presented languages are important in some way, but we emphasize:
- ⌘ WS-Agreement & WS-Policy as possible future general frameworks for WS QoS specification
  - ☒ Have industry support, but the "meat" is missing
- ⌘ WSLA as precise and detailed WS QoS specification language used in practice
  - ☒ Its solutions could be used (along with some ideas from other languages, e.g., WSOL) for the "meat"

Module IV of VI Slide 45 of 80

## Module IV: Overview of Approaches to WS QoS Management

- ⌘ Approaches to QoS monitoring
  - ☒ SOAP message intermediaries, probes, sniffers
- ⌘ Approaches to QoS discovery and selection
  - ☒ Using historical information vs. using contracts
  - ☒ Provider as only source of its QoS specifications, UDDI extensions, special QoS information registry
- ⌘ Some approaches to QoS control
  - ☒ Using different request queues for different QoS
  - ☒ Re-composition of Web services vs. re-negotiation of contracts

Module IV of VI Slide 46 of 80

## Using SOAP Message Intermediaries

```

    graph LR
      Consumer --> AP[Accounting Party]
      AP --> CEP[Constraint Evaluation Party]
      CEP --> MP[Measurement Party]
      MP --> Provider
      Provider -.-> MP
      MP -.-> CEP
      CEP -.-> AP
      AP -.-> Consumer
  
```

- ⌘ Exchange of monitored values: a) in SOAP headers; b) using special push or pull operations
- ⌘ Strengths: Realistic & consumer-specific measures
- ⌘ Weaknesses: High run-time overhead (can be reduced with periodic/occasional monitoring); results depend on network location of measurement

Module IV of VI Slide 47 of 80

## Using Probes (Probing)

```

    graph LR
      Consumer --> Provider
      Provider -.-> MP[Probe = Measurement Party]
      MP --> CEP[Constraint Evaluation Party]
      CEP --> AP[Accounting Party]
  
```

- ⌘ Strengths: Run-time overhead can be lower
- ⌘ Weaknesses: Results not consumer-specific, provider can treat probes in a special way; not possible to use SOAP headers to send monitored values; results depend on network location of probes

Module IV of VI Slide 48 of 80

## Using Sniffers (Sniffing)

```

    graph LR
      Consumer --> Provider
      Provider -.-> SN[Sniffer = Measurement Party]
      SN --> CEP[Constraint Evaluation Party]
      CEP --> AP[Accounting Party]
  
```

- ⌘ Strengths: Very low run-time overhead; measures can be realistic & consumer-specific
- ⌘ Weaknesses: Unknown SOAP message's Internet route; WS security technologies can be a problem; not possible to use SOAP headers to send monitored values; results depend on network location of sniffers

Module II of VI Slide 49 of 80

### Using Historical QoS Information for WS Selection - Possible Approaches

- ⌘ From the **same consumer**
  - ☒ Problem: When consumer did not previously invoke this operation of the provider Web service
- ⌘ From **probes**
  - ☒ Problem: Easy for providers to give excellent QoS to probes, while bad QoS to real consumers
- ⌘ From **all consumers**
  - ☒ Problem: Consumers have different characteristics (e.g., could be located on different continents)
  - ☒ Problem: Other consumers' reports can be fake

Module II of VI Slide 50 of 80

### Using Historical QoS Information for WS Selection - Discussion

- ⌘ General problem: **Circumstances of different invocations are different!**
  - ☒ Example: When the number of provider's concurrent consumers grows, it is likely that QoS perceived by individual consumers will drop
- ⌘ General problem: Absence of targets/goals to guide control activities (including billing)
- ⌘ Conclusion: Historical QoS information can be **useful**, but it provides **no guarantees** (and can even be misleading) => **contracts are needed**

Module II of VI Slide 51 of 80

### Using Provider as the Only Source of Its QoS Specifications

1. Provider publishes **only its WSDL file** to the registry (e.g., UDDI registry)
2. Consumer searches registry **only based on functionality** and selects one Web service
3. Consumer and provider **negotiate** QoS (and prices/penalties) that consumer will receive

- ⌘ Strengths: QoS can be **dynamic - registry not updated when QoS changes**; QoS can be customer-specific; no need to extend UDDI
- ⌘ Weaknesses: QoS not used for selection of WSes

Module II of VI Slide 52 of 80

### Using UDDI Extensions Describing QoS - Explanation and Some Options

1. Provider publishes **its WSDL file and QoS specifications** to the extended UDDI registry
2. Consumer searches registry **based on both functionality and QoS** and selects one WS (and its QoS - if multiple classes of service)
3. Consumer binds to the **provider and its QoS**

- ⌘ Option: Extended UDDI can be a gateway to the real UDDI, which is not changed
- ⌘ Option: [Ran2003] suggests a QoS certifier - probe verifying QoS guarantees in the registry

Module II of VI Slide 53 of 80

### Using UDDI Extensions Describing QoS - Discussion

- ⌘ Strengths: QoS used for selection of Web services; simple selection (not negotiation) of QoS
- ⌘ Weaknesses:
  - ☒ QoS changes require updating the registry (most works update only stored QoS specifications)
  - ☒ Consumers can have outdated QoS information (unless all are informed about updates, which requires keeping info about all consumer-provider relationships and has high run-time overhead)
  - ☒ If no classes of service - no QoS differentiation
- ⌘ Issue: Compatibility with the ordinary UDDI

Module IV of VI Slide 54 of 80

### Using Separate QoS Information Registry

```

graph TD
    Provider[Provider] -- "1. Publish WSDL" --> UDDI[UDDI Registry]
    Provider -- "2. Publish QoS" --> QoS[QoS Registry]
    Consumer[Consumer] -- "3. Find WSEs" --> UDDI
    Consumer -- "4. Find QoS" --> QoS
    Consumer -- "5. Select WS and its QoS" --> Provider
    Consumer -- "6. Bind" --> Provider
  
```

- ⌘ Several model variations (e.g., registry performs selection)
- ⌘ Strengths - same as for UDDI extensions, plus **no need to extend UDDI**
- ⌘ Weaknesses - same as for UDDI extensions, plus **higher run-time overhead and higher complexity**

Module IV of VI Slide 55 of 80

### Several Control Approaches That Try to Meet QoS Guarantees

1. Manipulate which request is processed first
  - ☒ Provider has several different request queues, e.g., one for each class of service
  - ☒ Scheduler within the provider decides from which queue to process a request, depending on QoS guarantees, current load, queue lengths, ...
2. Manipulate thread priorities for different requests and/or OS scheduling discipline
3. General approach: Manipulate allocation of resources for various requests
4. Load balancing between replicas

Module IV of VI Slide 56 of 80

### Re-composition of Web Services vs. Re-negotiation of Contracts

- ⌘ Run-time adaptation of WS compositions
  - a) Re-composition of Web services – more powerful
    - ☒ Special case: Switching between WSeS (only 1 change)
  - b) Re-negotiation of contracts – faster, simpler, lighter
    - ☒ Special case: Switching between classes of service

Legend: C – consumer; P – provider; CS – class of service

Module II of VI Slide 57 of 80

### Mechanisms for Run-Time Manipulation of Classes of Service

1. Switching between classes of service (CSeS)
  - ☒ Provider-initiated or consumer-initiated
2. Deactivation and reactivation of CSeS
  - ⌘ With accommodation of affected consumers
3. Deletion of deactivated CSeS
4. Creation of new CSeS (same WSDL files)
  - ☒ Provider decides when it is possible and allowed
5. Allowing/disallowing consumers to use a CS
  - ⌘ Not always a replacement for the alternatives (incl. re-negotiation of SLAs), but can be their simple, fast, and lightweight complement

Module IV of VI Slide 58 of 80

### Some QoS Issues at the SOAP Level

- ⌘ Reliable delivery of SOAP messages (e.g., "exactly once" delivery)
  - ☒ WS-Reliability - OASIS Web Services Reliable Messaging (WSRM) TC (originally: Fujitsu, Hitachi, NEC, Oracle, Sonic Software, Sun Microsystems)
  - ☒ WS-ReliableMessaging – OASIS Web Services Reliable Exchange (WS-RX) TC (originally: BEA Systems, IBM, Microsoft, TIBCO Software)
- ⌘ Compression of SOAP messages to reduce network transmission
  - ☒ Several approaches to XML compression

Module V of VI Slide 59 of 80

### Module V: Research and Industrial Tools for WS QoS Management

- ⌘ Published as research work in refereed papers
  - ☒ QoS-enabled selection of WSeS: UDDIe, UX
  - ☒ Contract-based QoS management: WSLA, WSMF ([SahaiEtAl2002]), WSOI (WSOL), WS-QoS, Cremona (WS-Agreement)
  - ☒ Other: Smartware, ... and many, many others
- ⌘ Commercial products (non-refereed literature)
  - ☒ WS QoS management products from specialized companies
  - ☒ Large systems management suites
  - ☒ Related management products

Module V of VI Slide 60 of 80

### UDDI Extensions: UDDIe and UX

- A. UDDIe (UDDI Extension)
  - from Cardiff Univ.; uses UDDIe data format (slide 42)
    - ☒ New operations for corresponding UDDIe search
    - ☒ Lease manager and lease-related operations
    - ☒ Used in G-QoS architecture for Grid computing
- B. UX (UDDI eXtension)
  - from Nanyang Tech.Univ.: C. Zhou, L.T. Chia, B.S. Lee
    - ☒ Acts as a gateway to the standard UDDI
    - ☒ Contains historical QoS information reported by consumers and probes ("test hosts")
    - ☒ QoS metrics defined implicitly

Module V of VI Slide 61 of 80

### Some Other Published Works on QoS-Enabled WS Discovery and Selection

- A. **Quality of Compliance (WS-QoC)** – monitors SLAs and calculates normalized weighted difference between QoS guarantees and monitored values from Cardiff Univ.: A. ShaikhAli, O.F. Rana, D. Walker
- B. **[Day & Deters 2004]** – consumers collect QoS info and report it to QoS info registry (“QoS forum”); rule-based and naive Bayes reasoners for WS selection from Univ. of Saskatchewan: J. Day, R. Deters
- C. **[DeoraEtAl2003]** – compare user ratings and expectations from Cardiff Univ.: V. Deora, J. Shao, W.A. Gray, N.J Fiddian

- ⌘ **Solution Management Service** – QoS info registry for [McGregor2003] extension of BPEL4WS (slide 42)

Module V of VI Slide 62 of 80

### Web Service Level Agreement (WSLA) Framework

from IBM Research; uses WSLA language (slides 26-28)

- ⌘ **Modules = services:** 1) Establishment; 2) Deployment; 3) Measurement; 4) Condition Evaluation; 5) Management; 6) Business Entity
  - ☒ Prototype: **SLA Compliance Monitor** – module 1 is simple, 2 is implemented, 3 & 4 are general purpose, 5 & 6 missing
  - ☒ Service Deployment Information (SDI) is a subset of WSLA
  - ☒ Special management port types (e.g., for value exchange)
- ⌘ **Strengths:** The most widely used WS QoS research infrastructure; comprehensive approach to QoS management; support for management third parties
- ⌘ **Weaknesses:** Run-time overhead

Module V of VI Slide 63 of 80

### Web Services Management Network (WSMN)

from HP Labs; uses [SahaiEtAl2002] language (slide 29)

- ⌘ **Main element: WSMN intermediary**
  - ☒ A proxy between the Web service and the outside
  - ☒ **Monitors and enforces SLAs** and exchanges management info using lifecycle, measurement, and assurance (e.g., negotiation) protocols
  - ☒ Run-time QoS info exchanged using protocols
  - ☒ General purpose modules within a SOAP engine
  - ☒ Prototype: **Business Management Platform (BMP) Agent** implemented over Apache SOAP (predecessor of Axis)
- ⌘ **Strengths:** Enables SLA-based QoS management
- ⌘ **Weaknesses:** Complexity & run-time overhead

Module V of VI Slide 64 of 80

### Web Service Offerings Infrastructure (WSOI)

from Carleton Univ.; uses WSOL language (slides 31-34)

- ⌘ Extends open-source **Apache Axis SOAP engine**, run over Apache Tomcat app server
- ⌘ **Monitoring** of WSOL service offerings
- ⌘ **Dynamic manipulation** of service offerings
- ⌘ **Strengths:** Unique management support for WS classes of service, relatively low run-time overhead
- ⌘ **Weaknesses:** Limited control activities to meet QoS guarantees, not all modules implemented in the prototype, incomplete documentation

Module V of VI Slide 65 of 80

### Web Service Offerings Infrastructure (WSOI) – Main Modules

- ⌘ **Modules for monitoring:**
  - ☒ Standard Axis modules (including handlers and chains)
  - ☒ **WSOI-specific handlers and chains (QoS monitoring)**
  - ☒ WSODEngine modules (for ordering of WSOI handlers)
  - ☒ Timer (for periodic activities)
- ⌘ **Modules for manipulation:**
  - ☒ **SOMgmtDecisions** (code of management algorithms)
- ⌘ **Modules for both monitoring & manipulation:**
  - ☒ Modules for **Service Offering Management (SOM)** port types (used in management protocols)
  - ☒ Data structures (service offering descriptions, consumer info, accounted management info within sessions, ...)

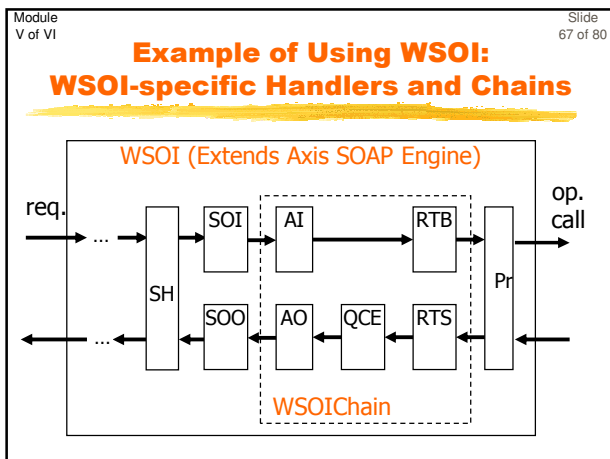
Module V of VI Slide 66 of 80

### WSOI Inside a Provider Web Service

- ☒ Can also be used in consumers and SOAP intermediaries
- ⌘ Exchange of management (incl. QoS) info: in SOAP headers or using special built-in push or pull operations (one of SOM port types)

```

graph LR
    subgraph Provider
        Tomcat
        WSOI["WSOI (extends Axis)"]
        Code["Code of operations"]
        Tomcat -- SOAP --> WSOI
        WSOI -- op. call --> Code
    end
    Consumer -- "SOAP over HTTP" --> Tomcat
  
```



Module V of VI Slide 68 of 80

### Web Services Quality of Service (WS-QoS) Infrastructure

from Freie Uni. Berlin; uses WS-QoS language (slide 35)

- ⌘ QoS-enabled WS discovery and selection
  - ☒ "QoS broker" – a variant of QoS info registry that (pre-)fetches QoS info and selects WSes and QoS
- ⌘ QoS monitoring – (mostly) by provider; "QoS channel" in SOAP headers = consumer sends metrics to measure, provider sends measured values
  - ☒ "QoS proxy" – between Web service and network
- ⌘ Strengths: Deals with both QoS-aware WS discovery/selection and QoS monitoring
- ⌘ Weaknesses: Assumes QoS-aware network layer

Module V of VI Slide 69 of 80

### CREation and MONitoring of Agreements (Cremona)

from IBM Research; uses WS-Agreement (slides 35-36)

- ⌘ Architecture for WS-Agreement middleware
  - ☒ Agreement initiator & agreement provider roles
  - ☒ Agreement Management layers: a) ... Protocol Role ... (APRM), b) ... Service Role ... (ASRM), c) Strategic ... (SAM)
- ⌘ Java library that: 1) implements WS-Agreement interfaces; 2) provides management functionality for agreement templates and instances; 3) defines abstractions to be implemented in domain-specific environments
- ⌘ Strengths: Relates agreements with underlying resources; reusable for various domains
- ⌘ Weaknesses: Needs additions to be used for WSes

Module V of VI Slide 70 of 80

### Smartware

from Infosys: A. Sharma, H. Adarkar, S. Sengupta

- ⌘ QoS control: Differentiated scheduling of requests based on context priorities
  - ☒ Context = info about provider application, user, and client device; sent by consumer in request SOAP header
- ⌘ Based on Apache Axis SOAP engine, adds:
  - ☒ Interceptor – reads context info and determines priority
  - ☒ Scheduler – puts request into a queue for its priority; based on scheduling policy fetches a request from a queue
  - ☒ Dispatcher – forwards request to the provider
- ⌘ Strengths: Rare work that performs QoS control
- ⌘ Weaknesses: Scheduling uses limited information

Module V of VI Slide 71 of 80

### Computational Quality Attributes (CQA) Processing Service

by C.K. Fung, P.C.K. Hung, R.C. Linger, G.H. Walton

- ⌘ Flow-Service-Quality (FSQ) engineering
- ⌘ Key component services: QoS Manager, Establishment S., Policy Manager, Resource Manager, Prediction S., Operation S., Maintenance S., Monitoring S., Adaptation S., Diagnostic S.
- ⌘ BPEL extended with WSLA expressed in CQA
- ⌘ Strengths: New approach to specification and evaluation of dynamic service- and flow-level constraints & QoS using Bayesian statistical models
- ⌘ Weaknesses: Monitoring is by probing only

Module V of VI Slide 72 of 80

### Some Other Published Works on QoS Management

- A. wsBus – WS middleware – intermediary between providers and consumers; provides message prioritization, reliability, fault tolerance, load balancing, and security from Univ. of New South Wales: A. Erradi, P. Maheshwari, ...
- B. [Yu & Lin 2005] – "QoS broker" with workflow & QoS information; implements dynamic switching of WSes from Univ. of California – Irvine: T. Yu, K.J. Lin
- C. Web Services Management Layer (WSML) – uses aspect-oriented programming to code WS monitoring from Vrije Univ. Brussel: B. Verheecke, M.A. Cibrán, V. Jonckers
- ⌘ Multichannel Adaptive Information Systems (MAIS) – not on WSes, but this adaptation can be applied to WSes from Univ. Roma & Pol. Milano: C. Marchetti, B. Pernici, P. Plebani

Module V of VI Slide 73 of 80

### Some Observations about Industrial Products for WS QoS Management

- ⌘ They address many practical problems
  - ☒ Academic researchers should be aware of these works and their accomplishments
  - ☒ Some works contain advanced solutions that show how SLAs and policies can be used in practice
- ⌘ Many products have significant limitations:
  - ☒ Crucial role of human administrators (i.e., not completely automated)
  - ☒ Limited/predefined choice of used QoS metrics
  - ☒ Lack of formal machine-understandable QoS specification (instead, forms are used)

Module V of VI Slide 74 of 80

### Some Products for WS QoS Management from Specialized Companies

- ☒ Often products (1 or more) addressing several management areas, including performance (QoS)
- ⌘ **Actional** SOA Command and Control (including SOAPStation Web Services Broker) – policies
- ⌘ **AmberPoint** (including Service Level Manager) – custom-made SLAs
- ⌘ **Blue Titan** (including Network Director) – policies
- ⌘ **Infravio** Ensemble (including X-broker) – custom-made Web Services Delivery Contracts (extend SLAs)
- ⌘ **WestGlobal** mScope (including Performance Management Module - PMM) – custom-made SLAs

Module V of VI Slide 75 of 80

### Large System Management Suites

- ⌘ Contain many different management products
  - ☒ some related to WSEs (or “business services”)
  - ☒ some related to performance (QoS) management of applications, computing systems, networks

1. HP: **OpenView** (includes **SOA Manager**)
2. IBM: **Tivoli** (includes Business Systems Manager)
3. Computer Associates (CA): **Unicenter** (includes **Web Services Distributed Management – WSDM**)
4. BMC Software: **Patrol** (includes MAINVIEW)
5. Microsoft (includes Application Center)

Module V of VI Slide 76 of 80

### Some Related Management Products

- ⌘ Web service infrastructures, such as:
  - ☒ **Grand Central Communications** Business Services Network
  - ☒ **IONA** Artix (very limited support for QoS)
  - ☒ **webMethods** Enterprise Services Platform
- ⌘ Application performance tools, such as:
  - ☒ Many tools use Java Management Extensions (JMX)
  - ☒ **Mercury**: Service Level Management, ...
  - ☒ **OPNET** (Altaworks): Commander, Panorama, ...
  - ☒ **Quest Software**: PerformaSure, Foglight, ...
- ⌘ Web service security management tools

Module VI of VI Slide 77 of 80

### Module VI: Summary of QoS Specification and Management for Web Services

- ⌘ QoS & manageability (also price, security, trust, ...) will be **differentiators** in the WS market
  - ☒ Determine WS to be chosen among similar ones
- ⌘ QoS specification is the basis for management
- ⌘ Management (monitoring & control) is necessary to meet QoS guarantees, discover and fix problems, accommodate change, ...
- ⌘ The vision of **service-oriented computing can not be achieved without (QoS) management**
  - ☒ But, the basic Web service technologies do not address QoS specification and management

Module VI of VI Slide 78 of 80

### The Main Achieved Results

- ⌘ Several ...
  - ☒ ... **languages** for precise and detailed specification of SLAs or classes of service for Web services
  - ☒ ... **general frameworks** that can be extended for QoS specification (WS-Policy, WS-Agreement, OWL-S)
  - ☒ ... **industrial standards** (WSDM, WS-Reliability, ...)
  - ☒ ... prototyped **solutions for QoS-enabled Web service discovery/selection**
  - ☒ ... research **infrastructures that enable WS QoS management** (predominantly monitoring)
  - ☒ ... **industrial products** used in practice

## Some Open Topics

- ☒ **Standards for WS QoS specification**
  - ☒ Contracts vs. policies; Extending WS-Policy, OWL-S, ...; Equal status of guarantees and requirements; ...
- ☒ **Automatic negotiation of QoS contracts**
- ☒ **Standards for WS QoS management interfaces**
  - ☒ WSDM, WS-Agreement, ...
- ☒ **Control of WSEs to meet contract guarantees**
  - ☒ Resource capacity planning and management; Building complex control plans; Trust/reputation management; ...
- ☒ **Integrated management of business operations, Web services, and underlying computing & communications infrastructure**
  - ☒ Standard models and mappings between them
- ☒ **Solutions for adaptation to various changes**

## Resources

- ⌘ **Publications** are scattered between many different conferences, journals, and books
  - ☒ Intl. Web Services Quality Workshop (WQW) at WISE (Web Information Systems Engineering) 2003, 2004
  - ☒ W3C Workshop on Constraints and Capabilities for Web Services, Oct. 2004 – input into future standardization
- ⌘ **Annotated bibliography available at:**  
<http://elab.njit.edu/vladimir/QoS4WS.html>
  - ☒ All works mentioned in this tutorial are listed
- ⌘ **Ask the tutorial presenters** (e-mail on Slide 1)
- ⌘ **Hire one of the tutorial presenters!** (Knowledge, experience, hard work, dedication, passion, ...)